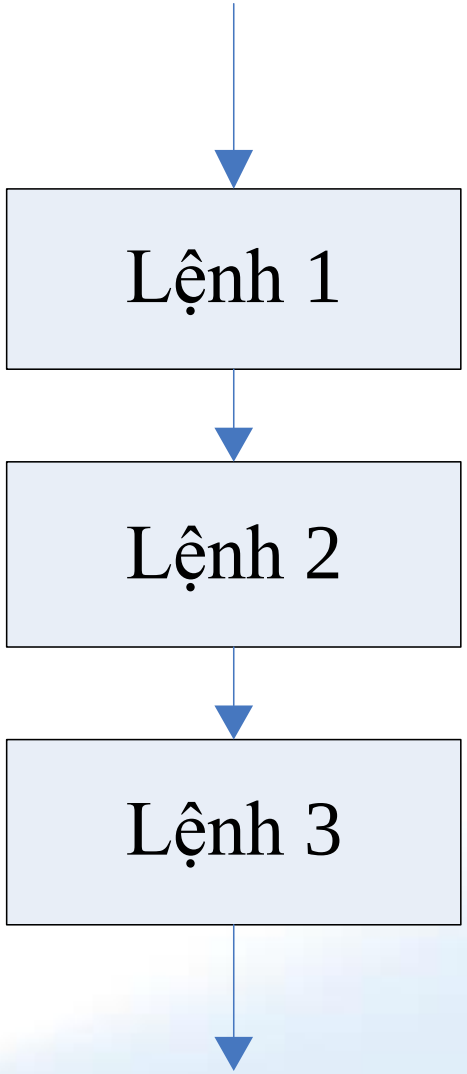
Chương 3
Các cấu trúc điều khiển

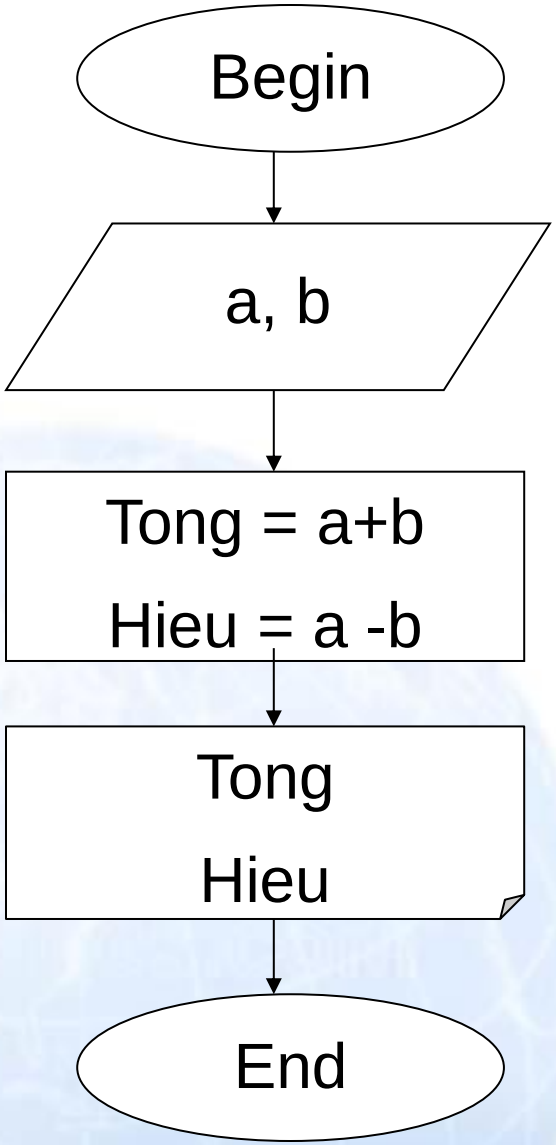
- Cấu trúc tuần tự
- Cấu trúc điều khiển rẽ nhánh
- Cấu trúc điều khiển lặp
- Một số thuật toán cơ bản
  - Thuật toán lặp tổng quát
  - Thuật toán tìm phần tử lớn nhất, phần tử nhỏ nhất
  - Thuật toán tìm ước số chung lớn nhất
  - Thuật toán kiểm tra số nguyên tố

Cấu trúc tuần tự

- Tuần tự thực thi tiến trình, mỗi lệnh được thực thi theo một chuỗi từ trên xuống, xong lệnh này rồi chuyển xuống lệnh kế tiếp.



❖Ví dụ: Nhập vào 2 số a,b. Tính tổng và hiệu



Cấu trúc điều khiển rẽ nhánh

- Cấu trúc rẽ nhánh chỉ cho máy tính chọn thực hiện một dãy lệnh nào đó dựa vào kết quả của một điều kiện (biểu thức quan hệ hay biểu thức so sánh)
  - **Gồm 2 dạng:**
  - **Chỉ xét trường hợp đúng**

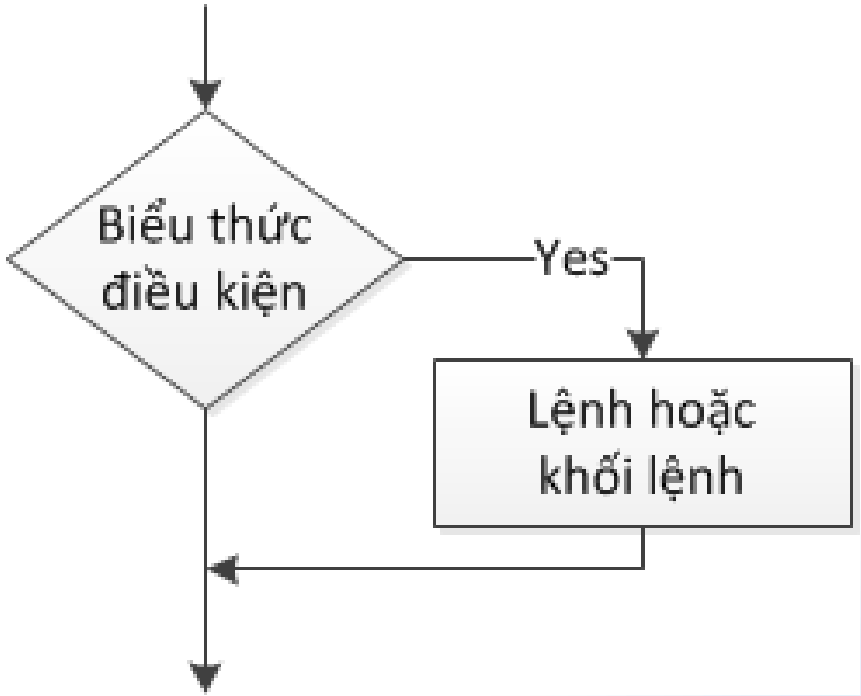
if *(biểu thức điều kiện)*

{

<khối lệnh> ;

}

Nếu biểu thức điều kiện cho kết quả ***true*** *thì thực hiện khối lệnh bên trong* ***if***



Ví dụ

❖Ví dụ: Tìm số lớn nhất trong hai số nhập từ bàn phím.

Thuật toán:

Khai báo biến a, b, max kiểu nguyên

Nhập giá trị cho hai biến a và b

Gán max =a // giả sử a là số lớn nhất

Nếu b > max thì max =b

In kết quả max.

❖ Chương trình:

Cấu trúc điều khiển rẽ nhánh

❑**Xét cả hai trường hợp đúng và sai:**

if *(biểu thức điều kiện)*

{

<khối lệnh 1>;

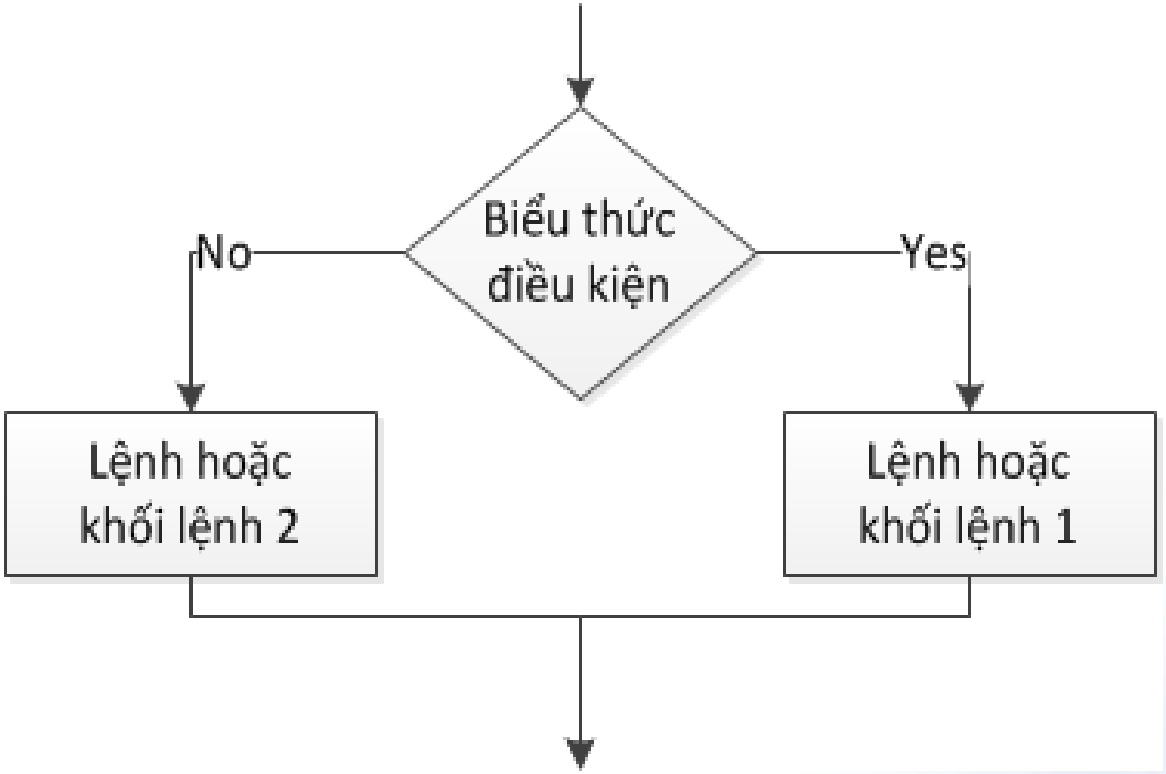
}

else

{

<khối lệnh 2>;

}



Nếu biểu thức điều kiện cho kết quả **true** thì thực hiện khối lệnh 1, ngược lại thì cho thực hiện khối lệnh thứ 2

Điều kiện khi dùng if

Các phép toán logic

>, >= , < , <=

== So sánh bằng

!= So sánh khác.

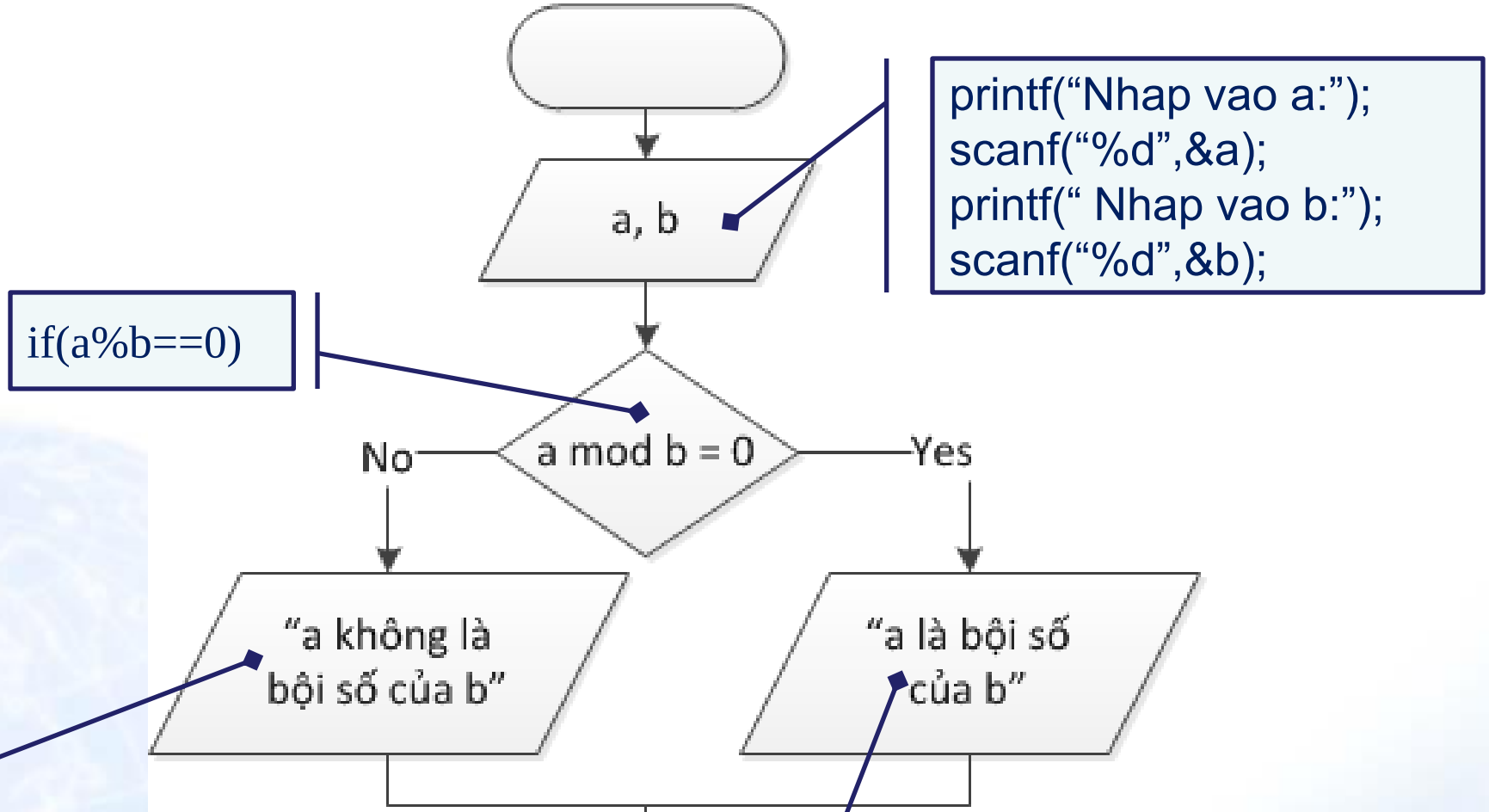
! phép phủ định

Ví dụ:

if (a > b)

if(a!=b)

Ví dụ: Nhập vào số nguyên a và b, nếu a là bội số của b thì in thông báo "a là la boi so cua b", ngược lại in "a khong la boi so cua b"



9

❖ Cài đặt:

Khi có nhiều hơn 1 điều kiện

- Phép toán Và (&&)

Là ĐÚNG khi tất cả điều kiện đưa vào là đúng.

A	B	KQ
1	1	1
1	0	0
0	1	0
0	0	0

Khi có nhiều hơn 1 điều kiện

❖ Phép toán Hoặc (||)

Là SAI khi tất cả điều kiện đưa vào là SAI.

A	B	KQ
1	1	1
1	0	1
0	1	1
0	0	0

Ví dụ

- Nhập vào điểm Toán, Lý, Hoá. Tính ĐTB, sau đó xét ĐTB

Nếu DTB>=8 thì xếp loại giỏi

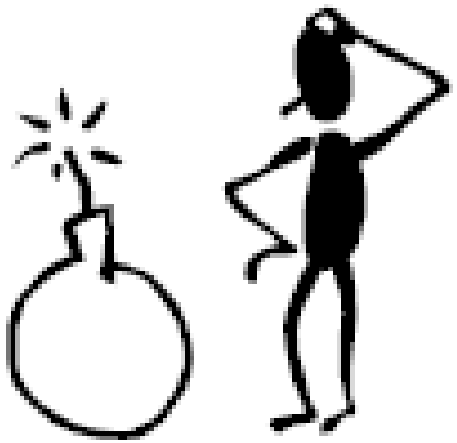
Nếu 8 > ĐTB >=5 thì xếp loại khá

còn lại là trung bình

```
DTB = (T+L+H)/3;
if (DTB>=8)
        printf("Gioi");
else
        if (DTB>=5) && (DTB<8)
                printf("Kha");
        else
                printf("TB");
```

Nhầm lẫn khi dùng if

```
#include <stdio.h>
void main()
{
    clrscr();
    int     number;
    printf("Nhap vao mot so nguyen duong:");
    scanf("%d",&number);
    if (number % 2=0)
       printf("%d la so chan\n",number);
    else
       printf("%d la so le\n",number);
}
```



- Chương trình trên sai ở đâu?

Chú ý khi dùng if-else

- Câu lệnh **if-else** lồng nhau
- **else sẽ kết hợp với if gần nhất chứa có else**
- **Trong trường if bên trong không có else thì phải viết nó trong cặp dấu {} để tránh sự kết hợp else if sai.**
- **Ví dụ:**

  giả sử biến so1 =3, so2=5, so3=10, a = 1

```
if (so1>0)
        if (so2 > so3)
                a = so2;
else
        a = so3;
```

```
if (so1>0)
{       if (so2 > so3)
                a = so2;
}
else
        a = so3;
```

Kết quả?????

Cấu trúc lựa chọn (switch…case)

```
switch (biểu thức)
{
    case n1 :
            các câu lệnh ;
            break ;
    case n2:
            các câu lệnh ;
            break ;
     ……
    case nk :
            <các câu lệnh> ;
            break ;
    [default:    các câu lệnh]
}
```

Trường hợp giá trị biểu thức bằng n_1

Trường hợp giá trị biểu thức bằng n2

Trường hợp giá trị biểu thức bằng nk

Các trường hợp còn lại (ko bắt buộc)

Với:

- **n_i**: các hằng số nguyên hoặc ký tự.
- Nếu giá trị của **biểu thức** = **n_i** $\Rightarrow$ thực hiện câu lệnh sau case **n_i**.
- Nếu giá trị biểu thức khác tất cả các giá trị n_i $\Rightarrow$ thực hiện câu lệnh sau **default** nếu có hoặc thoát khỏi **switch**.
- Khi chương trình đã thực hiện xong câu lệnh của case **n_i** nào đó thì nó sẽ thực hiện luôn các lệnh thuộc case bên dưới nó mà không xét lại điều kiện (do các **n_i** được xem như các nhãn)

→ Vì vậy, để chương trình thoát khỏi lệnh **switch** sau khi thực hiện xong một trường hợp, ta dùng lệnh **break**.

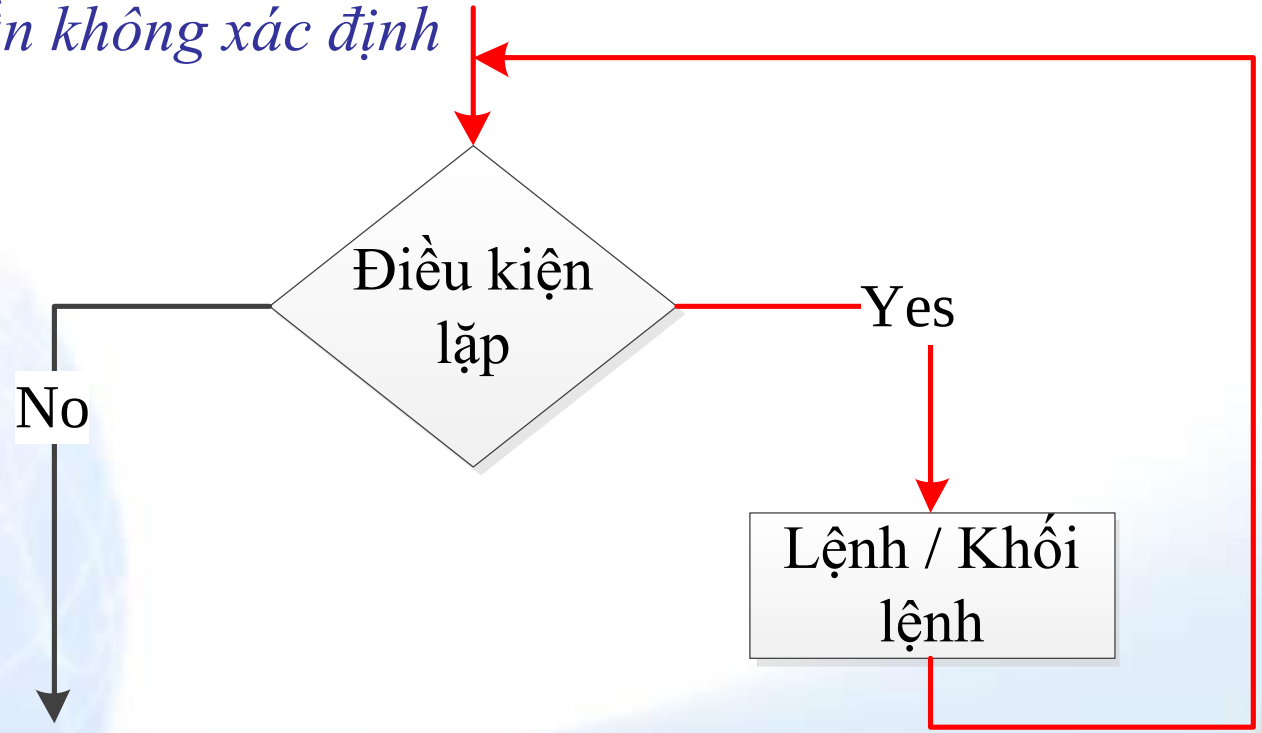
Ví dụ

In ra màn hình học lực của học sinh theo thang điểm như sau: Từ 0 -> 3: Kém, 4: Yếu, 5-> 6: Trung bình, 7 -> 8: Khá, 9 -> 10: Giỏi.

```
switch(diem)
      {
            case 0: case 1: case 2: case 3:
                  printf("Kem\n"); break;
            case 4: printf("Yeu\n"); break;
            case 5: case 6:
                  printf("Trung binh\n"); break;
            case 7: case 8: printf("Kha\n"); break;
            case 9: case 10:
                  printf("Gioi\n"); break;
            default: printf("Nhap diem sai\n");
      }     // Kết thúc switch
```

Cấu trúc lặp

- Cho phép lặp lại thực hiện 1 công việc nhiều lần.
- Có 2 loại:
  - *Lặp với số lần xác định*
    - **for**
  - *Lặp với số lần không xác định*
    - **while**
    - **do-while**



Vòng lặp for

for (<khởi gán> ; <điều kiện lặp> ; <cập nhật>)

{

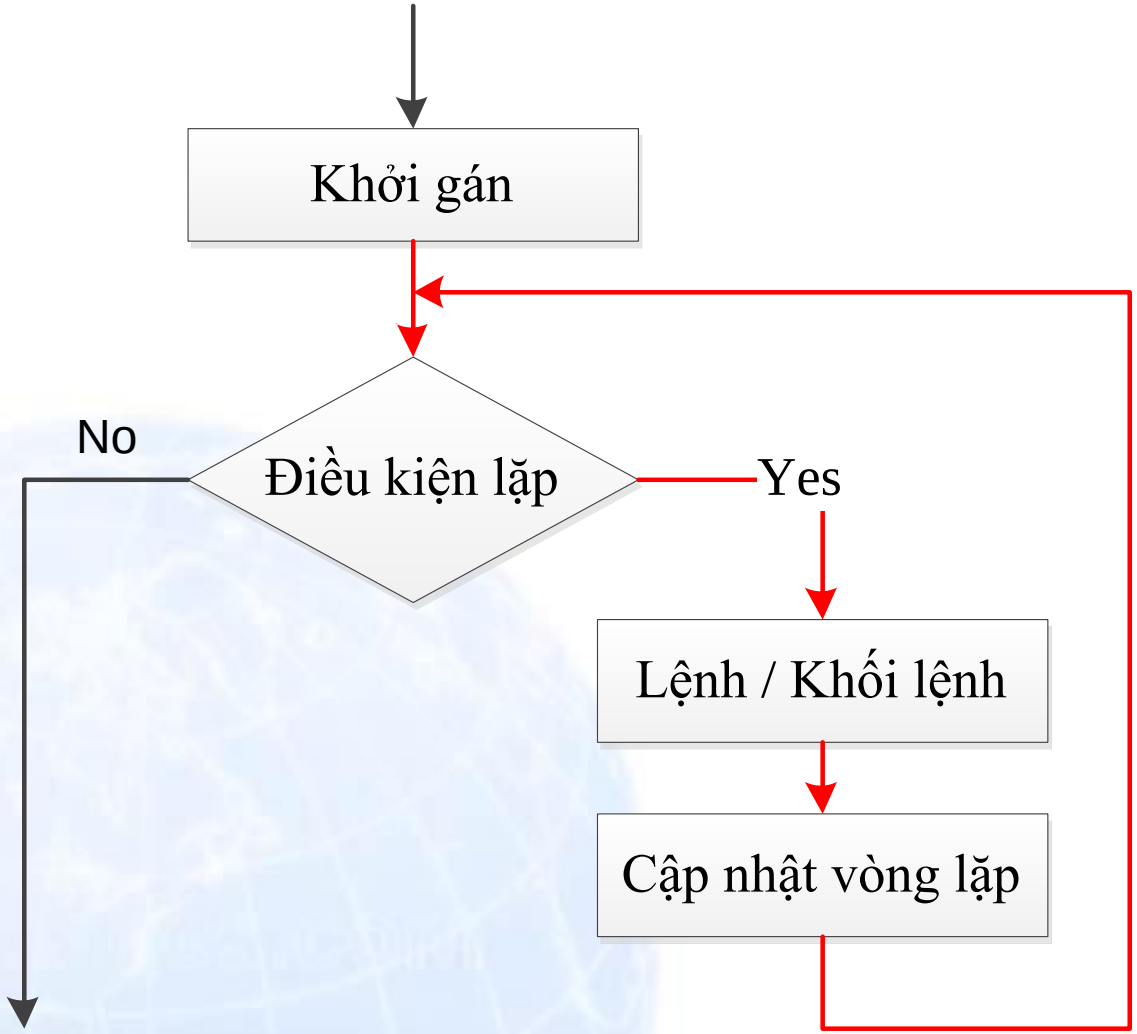
<khối lệnh>;

}

- *Khởi gán: Dùng để khởi gán giá trị ban đầu cho vòng lặp*
- *Điều kiện lặp: Dùng để kiểm tra điều kiện trước khi thực hiện vòng lặp*
- *Cập nhật: Dùng để cập nhật vòng lặp (tăng hoặc giảm chỉ số lặp)*

Bất kỳ biểu thức nào trong 3 biểu thức nói trên đều có thể vắng nhưng phải giữ dấu chấm phẩy (;)

Hoạt động



- ***Bước 1:*** *Khởi gán*
- ***Bước 2:*** *Kiểm tra điều kiện*

  *- Nếu **điều kiện bằng true** thì cho thực hiện các lệnh của vòng lặp, thực hiện **cập nhật vòng lặp**. Quay trở lại bước 2.*

  - Ngược lại thoát khỏi lặp.

Ví dụ

- Nhập vào một số nguyên dương.Xuất ra số từ 1->n.

Ví dụ

❖ Nhập vào một số nguyên dương.Tìm các ước số của nó.

Vòng lặp while

<Khởi gán>

while (<điều kiện lặp>)

{

lệnh/ khối lệnh;

<cập nhật>

}

- *Ý nghĩa:* Nếu giá trị của *điều kiện lặp* còn khác 0 (còn đúng) thì còn thực hiện *lệnh/ khối lệnh*.

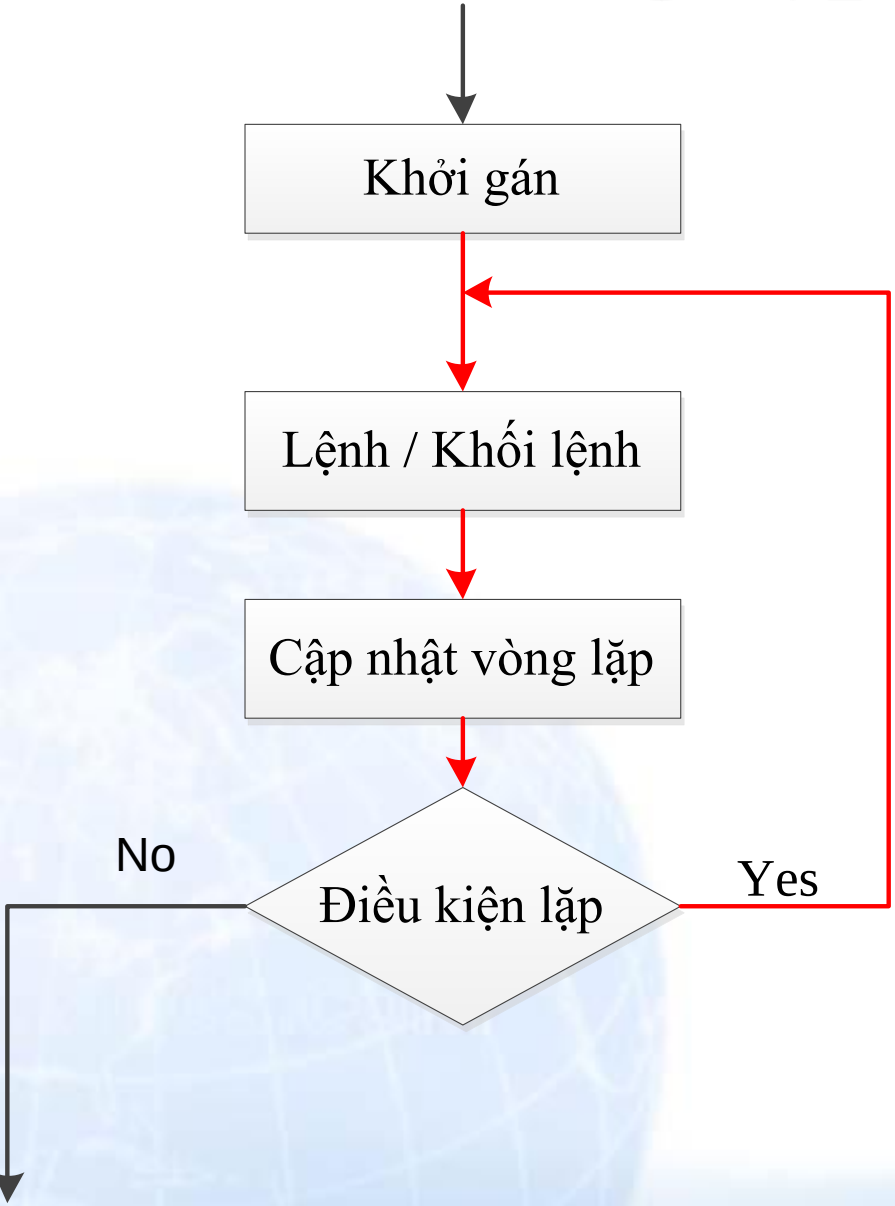
Vào thân vòng lặp ít nhất 0 lần.

Lưu ý: Cách hoạt động của while giống for

Ví dụ

- Nhập số nguyên n. In ra dãy số số nguyên từ 1..n

Vòng lặp do … while



< Khởi gán>

do

{

<khối lệnh>;

<cập nhật>;

*} **while** (điều kiện);*

Ý nghĩa:

Thực hiện <*khối lệnh*> cho đến khi giá trị của *điều kiện* bằng 0 (sai) thì dừng.

Vào thân vòng lặp ít nhất 1 lần

Ví dụ

Nhập số nguyên n. Tính tổng cho tới khi n = 0 thì dừng

So sánh các vòng lặp

- *Vòng lặp for/while:*
  - *Kiểm tra điều kiện trước thực hiện công việc sau.*
  - *Công việc có thể không được thực hiện lần nào.*
  - *Vòng lặp kết thúc khi nào điều kiện sai.*
- *Vòng lặp do-while*
  - *Thực hiện công việc trước kiểm tra điều kiện sau.*
  - *Công việc được thực hiện ít nhất 1 lần.*
  - *Vòng lặp kết thúc khi nào điều kiện sai.*

Câu lệnh đặc biệt

❖ Lệnh **return**

- *Khi gặp lệnh return máy sẽ kết thúc hàm chứa nó.*

❖ Lệnh **break**

- *Dùng để thoát khỏi vòng lặp **for, while, do…while** hoặc **switch-case**.*
- *Tiếp tục thực hiện lệnh bên ngoài sau vòng lặp hoặc **switch-case** nếu có*

Cần phân biệt với lệnh **return** là lệnh trả về từ hàm, nghĩa là thoát khỏi hàm đang thi hành, nên cũng giúp thoát luôn khỏi tất cả các vòng lặp.

Câu lệnh đặc biệt

❖ Lệnh **continue**

- *Trong vòng lặp, khi gặp lệnh **continue**, chương trình sẽ bỏ qua các câu lệnh sau **continue***
  - **for:** *quay lên tính trị cho biểu thức, rồi kiểm tra điều kiện coi có lặp tiếp không.*
  - **while/do-while:** *kiểm tra điều kiện coi có lặp tiếp không.*

❖ Lệnh **goto**

➢ ***Cú pháp khai báo***

```
goto nhãn;
```

✓ *Khi gặp lệnh goto máy sẽ nhảy tới thực hiện câu lệnh viết sau **nhãn**.*

Ví dụ

In ra màn hình giá trị từ 10 đến 20 trừ đi số 13 và số 17.

```
#include <stdio.h>
#include <conio.h>
void main()
{
        for (int k = 10; k <= 20; k++)
        {
            if (k == 13 || k == 17)
                continue; //tiếp tục thực hiện for ko thực hiện câu lệnh bên dưới
            printf("%d\t",k);

        }
        getch(); //dừng màn hình xem kết quả
}
```

Một số thuật toán lặp

❖Thuật toán lặp tổng quát:

Thuật toán lặp tổng quát có dạng:

- *Lặp i = 1, 2,..., n làm*
  - *Gọi một thủ tục xử lý cho lần lặp thứ i*
- *Cuối lặp*

Thuật toán tìm ước số chung lớn nhất

- Vấn đề :Viết chương trình nhập từ bàn phím hai số a, b. In ra màn hình ước số chúng lớn nhất.
- Xác định bài toán

Input: 2 số a và b nguyên dương

Output: USCLN(a, b)

Thuật toán tìm ước số chung lớn nhất

❖ Thuật toán

❑ Bằng mã giả:

Bước 1: Kiểm tra a và b

Nếu a khác b thì

Nếu a > b thì a = a - b, quay lại bước 1

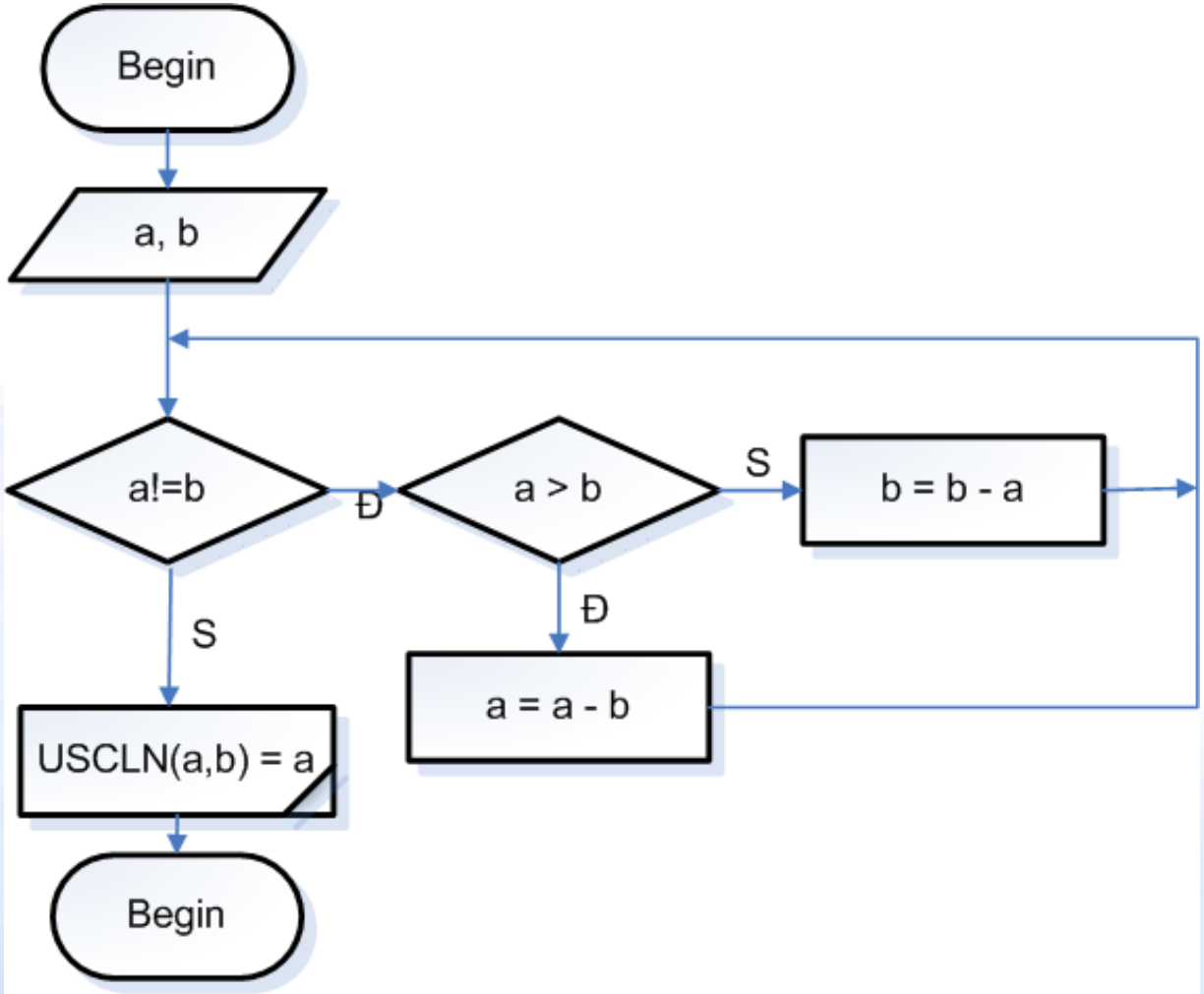
Nếu a <b thì b = b – a, quay về bước 1

Ngược lại (a=b) chuyển về bước 2

Bước 2: In kết quả và kết thúc

Thuật toán tìm ước số chung lớn nhất

- Lưu đồ giải thuật



Tìm ước số chung của 2 số nguyên dương a, b

Thuật toán kiểm tra số nguyên tố

- Định nghĩa: *Số nguyên tố là số tự nhiên chỉ chia hết cho 1 và chính nó. Ví dụ: 2, 3, 5, 7...*
- Vấn đề : Cho một số nguyên dương n. Kiểm tra n có phải là số nguyên tố hay không?
- Xác định bài toán:
  - *Input: n nguyên dương*
  - *Output: kết luận về tính nguyên tố n*
- Thuật toán:

Thuật toán kiểm tra số nguyên tố

1. Nếu n <= 1 thì
   {
   Kết luận: n không nguyên tố
   Dừng thuật toán
   }
2. flag = 1; //gán cho cờ flag giá trị đúng = 1 TRUE
3. Lặp i = 2, 3, 4,…., n-1
   Nếu (i là ước số của n) thì
   {
   flag = 0; //gán cho cờ flag giá trị sai = 0 FALSE
   Thoát khỏi vòng lặp
   }
4. Nếu flag = 1 thì
   Kết luận: n là số nguyên tố
   Ngược lại
   Kết luận: n không là số nguyên tố

Kiểm tra n có phải là số nguyên tố ?

Thuật toán kiểm tra số nguyên tố

- Nếu số nguyên n >1 không phải là một số nguyên tố thì n có một ước số nguyên tố (dương) <= $\sqrt{n}$
- Cải tiến thuật toán : Cho lặp i = 2,3,…$\sqrt{n}$
- Tức là:

```
for (int i=2; i<= sqrt(n); i++)
```

Nhập một số nguyên dương n. Xuất ra số ngược lại. Ví dụ:

Nhập 1706 ➔ Xuất 6071

Bài tập

Bài tập

Bài tập